GT-MOF (Movilidad con OpenFlow)

Coordinadora Liane Tarouco (UFRGS/RNP)
Colaborador Leandro Bertholdo (UFRGS/RNP)

15th Reunión Técnica de CLARA
7 - 11 de Noviembre de 2011
Montevideo - Uruguay

This project is funded by the European Union

A project implemented by CLARA

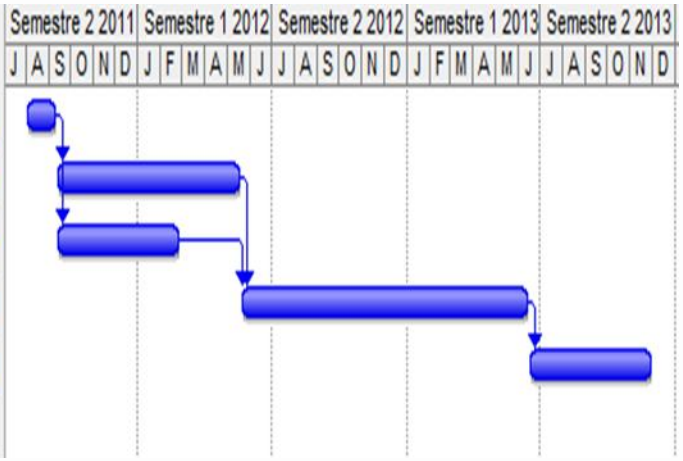
Resumen

- Objetivo
- Cronograma
- Tecnologia Openflow
  - El protocolo OpenFlow
  - El controlador NOX
- El Software mininet
- Prueba realizada

Objetivo y Cronograma

- El **GT-MOF** promoverá la investigación y el desarrollo de una solución capaz de ofrecer movilidad a los usuarios de una red Wi-Fi usando **tecnologías como OpenFlow** Wireless.

	Nome da tarefa	Duração	Início	Término
1	Estudo da Tecnologia OpenFlow Wireless	30 dias	Seg 01/08/11	Sex 09/09/11
2	Preparação do Ambiente	180 dias	Seg 12/09/11	Sex 18/05/12
3	Definição do Modelo de Testes e Medições	120 dias	Seg 12/09/11	Sex 24/02/12
4	Desenvolvimento e Implementação da Solução	280 dias	Seg 21/05/12	Sex 14/06/13
5	Testes de Performance e Funcionalidade	120 dias	Seg 17/06/13	Sex 29/11/13



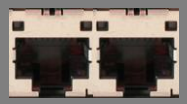
¿Cómo OpenFlow trabaja?

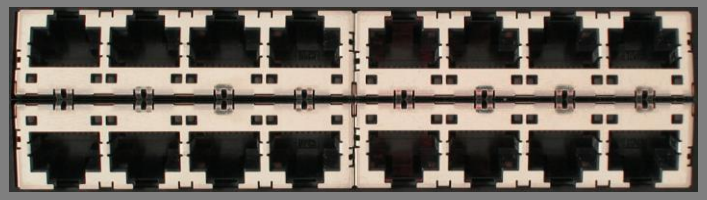
15th Reunión Técnica de CLARA
7 - 11 de Noviembre de 2011
Montevideo - Uruguay

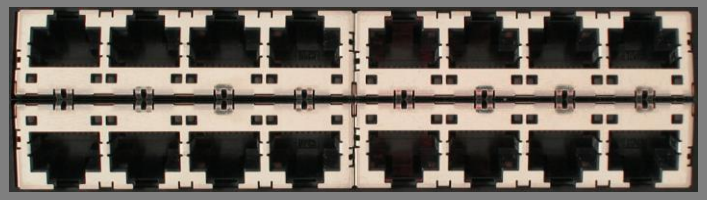
This project is funded by the European Union
A project implemented by CLARA

Estrutura interna de un switch

Ethernet Switch





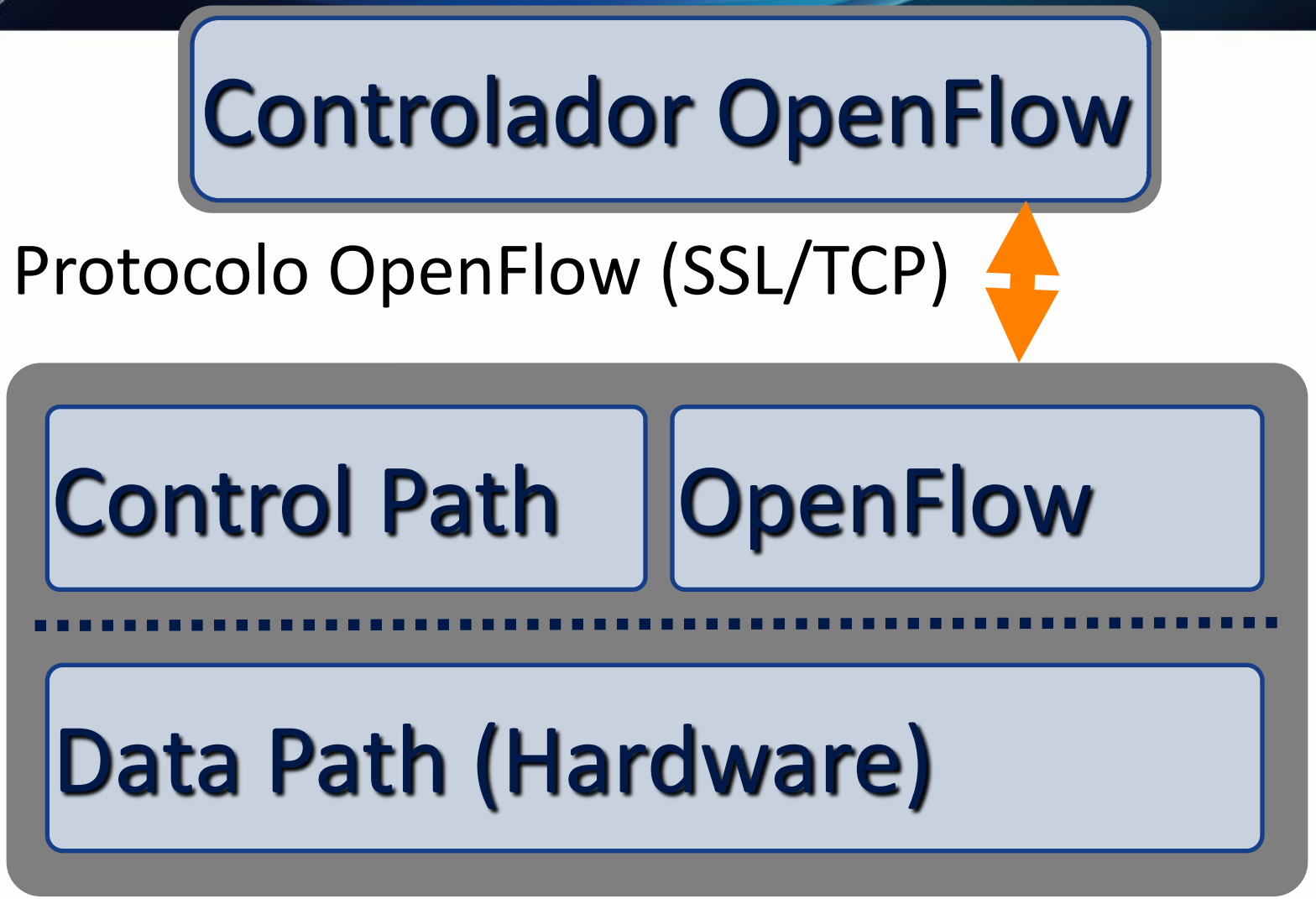


Estrutura interna de un switch

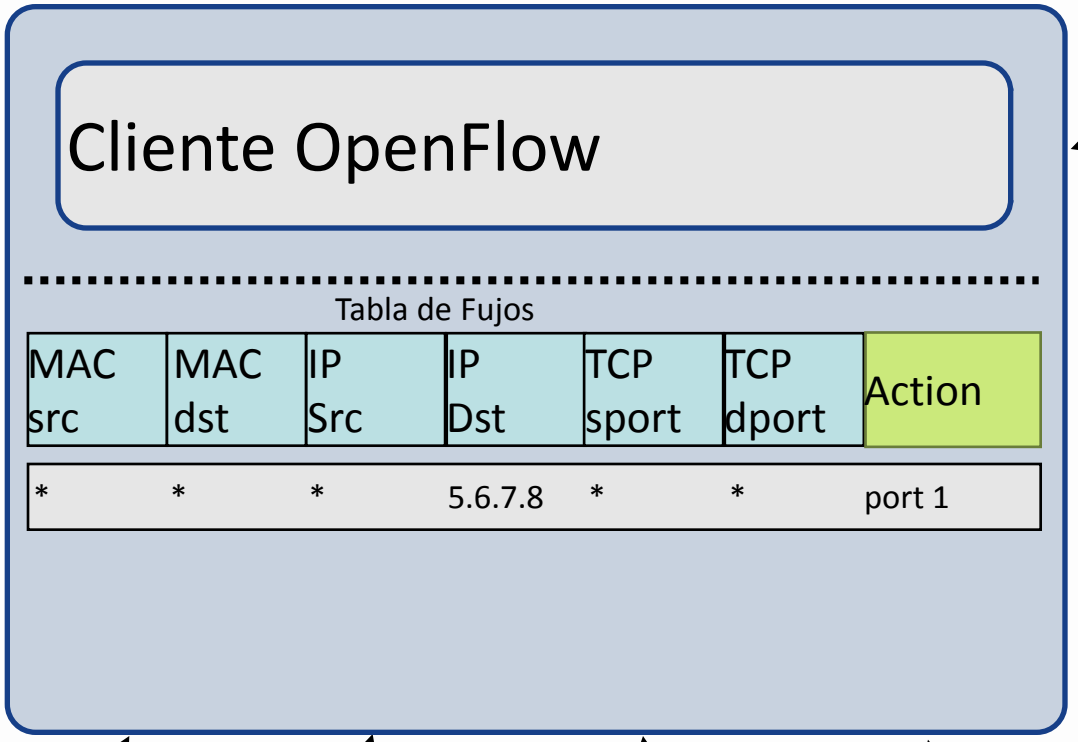
Control Path (Software)

Data Path (Hardware)

Estrutura interna de un switch



Ejemplo de Openflow



Controlador

port 1 port 2 port 3 port 4

5.6.7.8 1.2.3.4

OpenFlow Basico

Las entradas de la tabla de flujos

Regla	Acción	Estadísticas

Paquete+ contadores de bytes

1. Encaminar paquete para cero (ningúna) o más puertas
2. Encapsular y transmitir al controlador
3. Envia a canalización de procesamiento normal
4. Modifica campos
5. **Las extensiones que añadir!**

Switch Port	VLAN ID	VLAN pcp	MAC src	MAC dst	Eth type	IP Src	IP Dst	IP ToS	IP Prot	L4 sport	L4 dport

+ máscara que coincida con los campos

Ejemplos

Conmutación (Switching)

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Acion
*	*	00:1f:..	*	*	*	*	*	*	*	port6

Conmutación de flujos (Flow Switching)

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
port3	00:20..	00:1f..	0800	vlan1	1.2.3.4	5.6.7.8	4	17264	80	port6

Cortafuego (Firewall)

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	*	*	*	22	drop

Ejemplos

Enrutamiento (Routing)

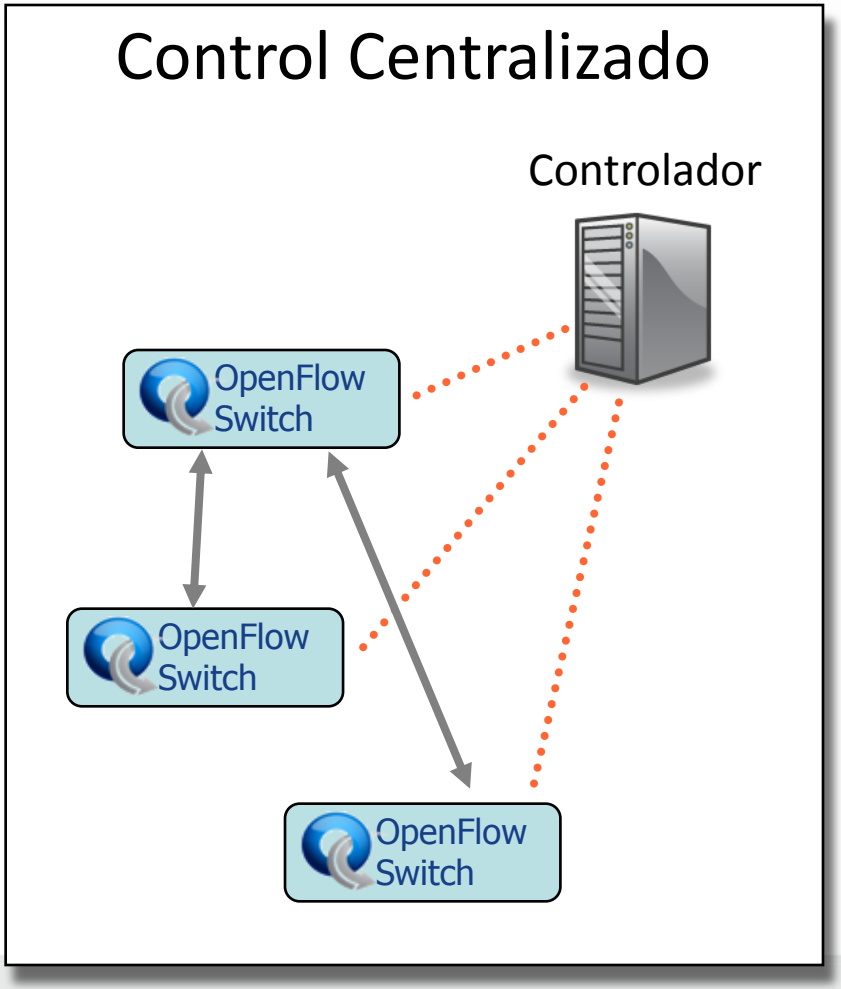
Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	5.6.7.8	*	*	*	port6

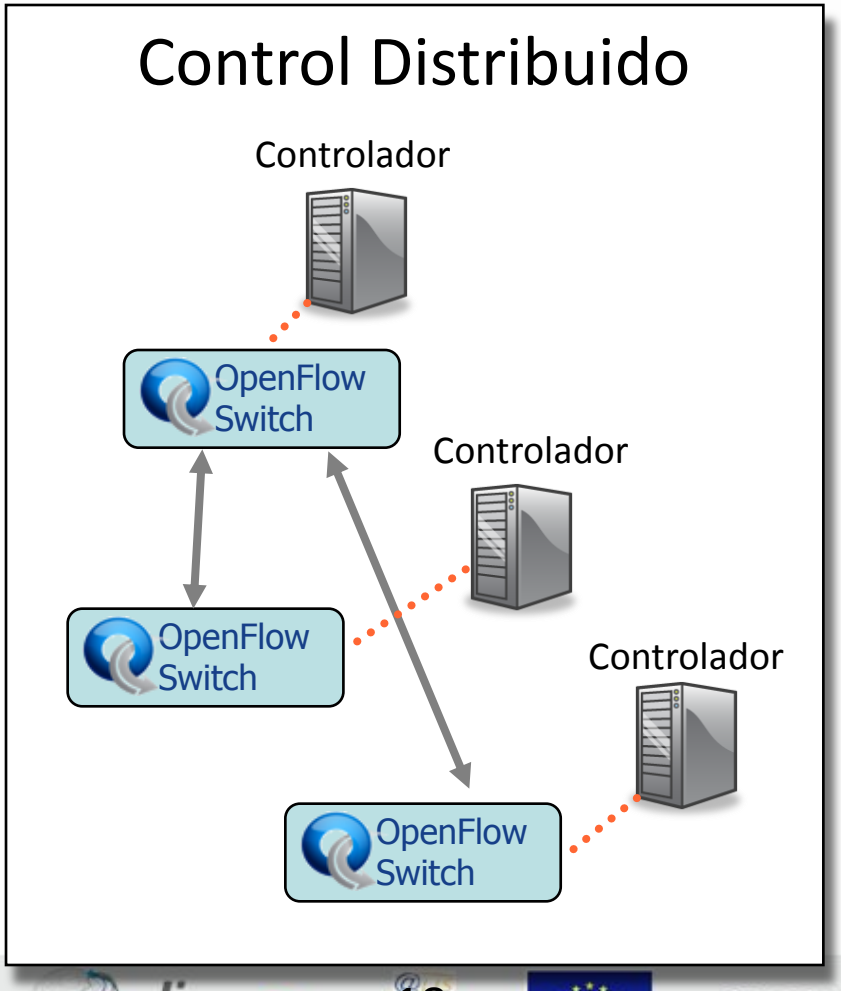
Conmutación de VLANs (VLAN Switching)

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f..	*	vlan1	*	*	*	*	*	port6, port7, port9

Control Centralizado vs Distribuido

Ambos los escenarios son posibles con OpenFlow





Enrutamiento de flujo vs. Agregación

Ambos los escenarios son posibles con OpenFlow

Baseado en flujo

- Cada flujo individual es creado por el controlador
- Coincidencia exacta de las entradas de flujo
- Tabla de flujo contiene una entrada por flujo
- Bueno para el control de grano fino, por ejemplo, las redes de campus

Agregado

- Una entrada de flujo abarca un gran grupo de flujos
- Entrada de flujo de comodín
- Flow table contains one entry per category of flows
- La Tabla de flujo contiene una entrada por cada categoría de flujos
- Bueno para un gran cantidad de flujos (ej. backbone)

Reactivo vs. Proactivo (pre-poblada)

Ambos los escenarios son posibles con OpenFlow

Reactivo

- El Primer paquete del flujo dispara la inserción del flujo en el controlador
- Uso eficiente de la tabla de flujo
- Cada flujo incurre en tiempo adicional de configuración de flujo
- Si perde la conexión de control, el conmutador tiene una utilidad limitada

Proactivo

- Controlador rellena previamente la tabla de flujo
- No hay pierda de tempo para configuracion de la tabla
- Pérdida de control de la conexión no interrumpe el tráfico
- Requiere reglas esencialmente agregadas (comodín)

Que **no se puede hacer** con OpenFlow v1.0

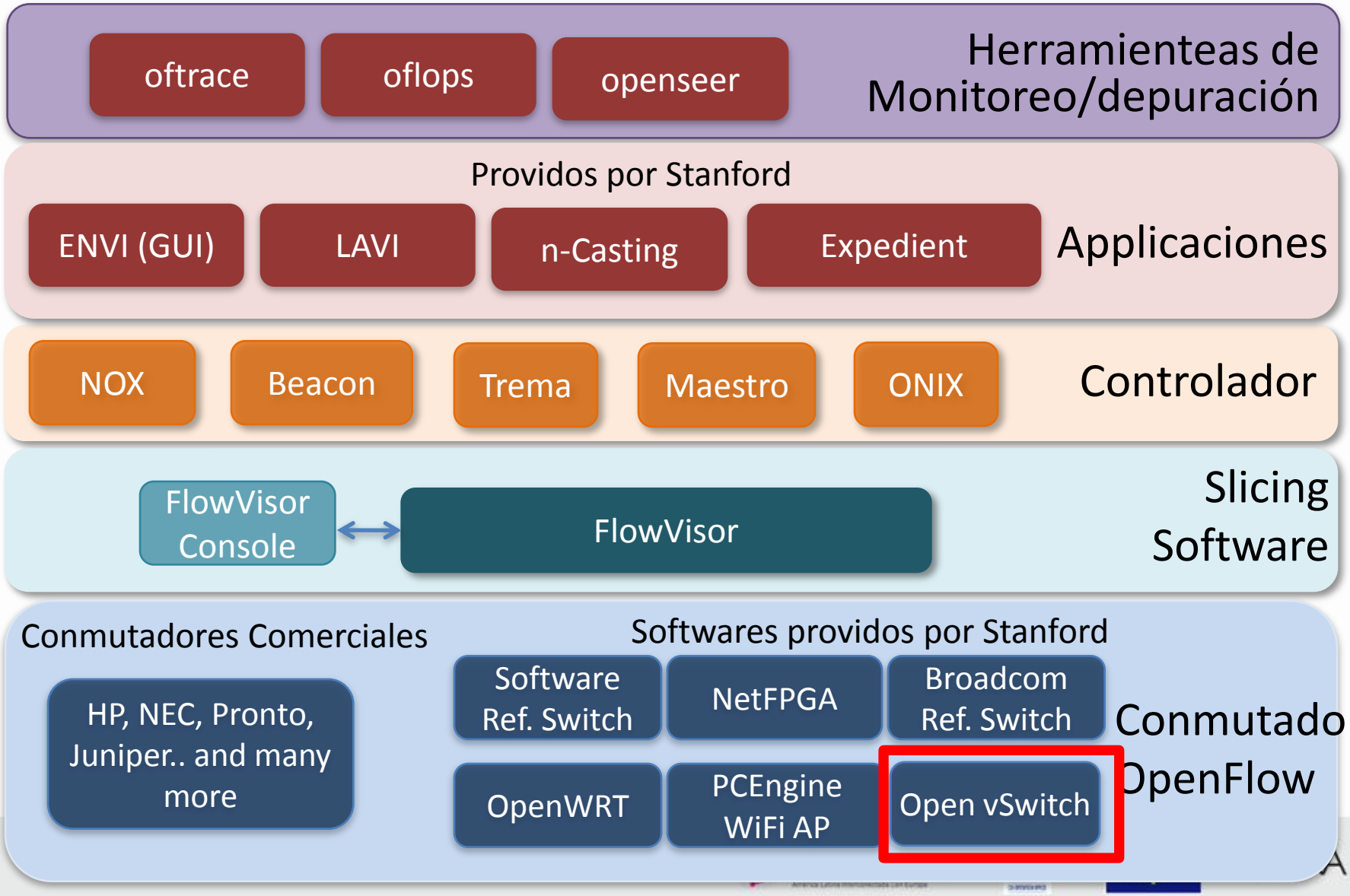
- Redes no baseada en flujos (per-paquete)
  - ej. Seleccion del próximo salto per-paquete
  - PERO OpenFlow puede proporcionar la instalación de cañerías para conectar estos sistemas
- Utilizar todas las tablas de flujo del chip
  - Si, és una limitación importante (cross-product issue)
  - Pero OF version 1.1 expone a estos, proporcionando una manera de los evitar
- Redes Ópticas y otro limites
  - Depende de nuevas versiones (OF 1.2, OF 2.0)

Bloques de Construcción

15th Reunión Técnica de CLARA
7 - 11 de Noviembre de 2011
Montevideo - Uruguay

This project is funded by the European Union
A project implemented by CLARA

Bloques de Construcción



Hardware que soporta SDN (Redes Definidas por Software)

Juniper MX-series

NEC IP8800

WiMax (NEC)

HP Procurve 5400

Netgear 7324

PC Engines

Pronto 3240/3290

Ciena Coredirector

Controladores de codigo livre

Nombre	Lang	Platforma	Licencia	Autor	Nota
OpenFlow Reference	C	Linux	OpenFlow License	Stanford/Nicira	no está diseñado para la extensibilidad
NOX	Python, C++	Linux	GPL	Nicira	desarrollando activamente
Beacon	Java	Win, Mac, Linux, Android	GPL (core), FOSS Licenses for your code	David Erickson (Stanford)	tiempo de ejecución modular, framework web
Maestro	Java	Win, Mac, Linux	LGPL	Zheng Cai (Rice)	
Trema	Ruby, C	Linux	GPL	NEC	Incluie emulador para testes
RouteFlow	?	Linux	Apache	CPqD (Brazil)	**enrutamiento IP virtual como un servicio (quagga)**

Resúmen

15th Reunión Técnica de CLARA
7 - 11 de Noviembre de 2011
Montevideo - Uruguay

This project is funded by the European Union
A project implemented by CLARA

Resúmen

- OpenFlow és un protocolo
  - Cómo se usa depende de ti
- SDN (Software-Defined Networking) és una arquitectura.
  - OpenFlow es sólo una pieza de ella…
  - OpenFlow está disponible, és usable y está mejorando
- **Estas son las primeras etapas de SDN, OF (OpenFlow) y ONS (Optical Network Service)**

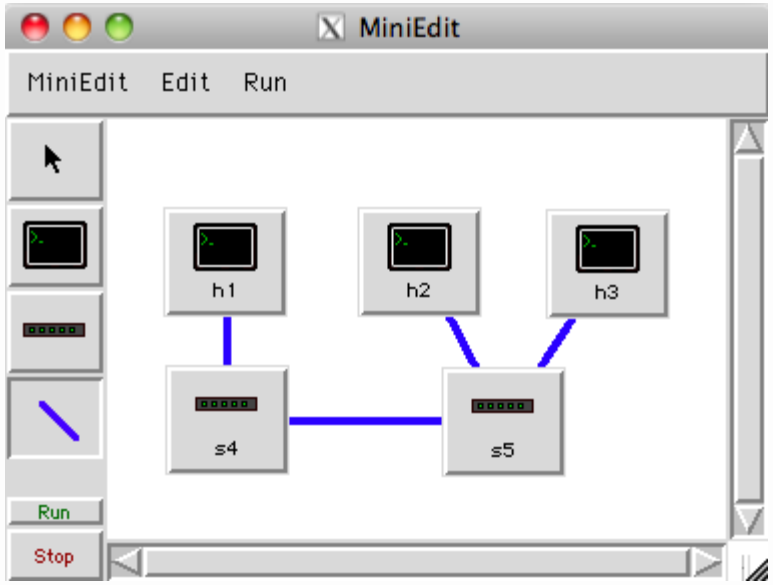
Ejemplo de test con Mininet

15th Reunión Técnica de CLARA
7 - 11 de Noviembre de 2011
Montevideo - Uruguay

This project is funded by the European Union
A project implemented by CLARA

Mininet

- Plataforma escalable para emular virtualización con OpenFlow
  - Puede tener hasta cientos de nodos, dependiendo de la configuración
  - Solo necesita una PC
  - Virtualización leve en Linux
  - Bueno para crear prototipos



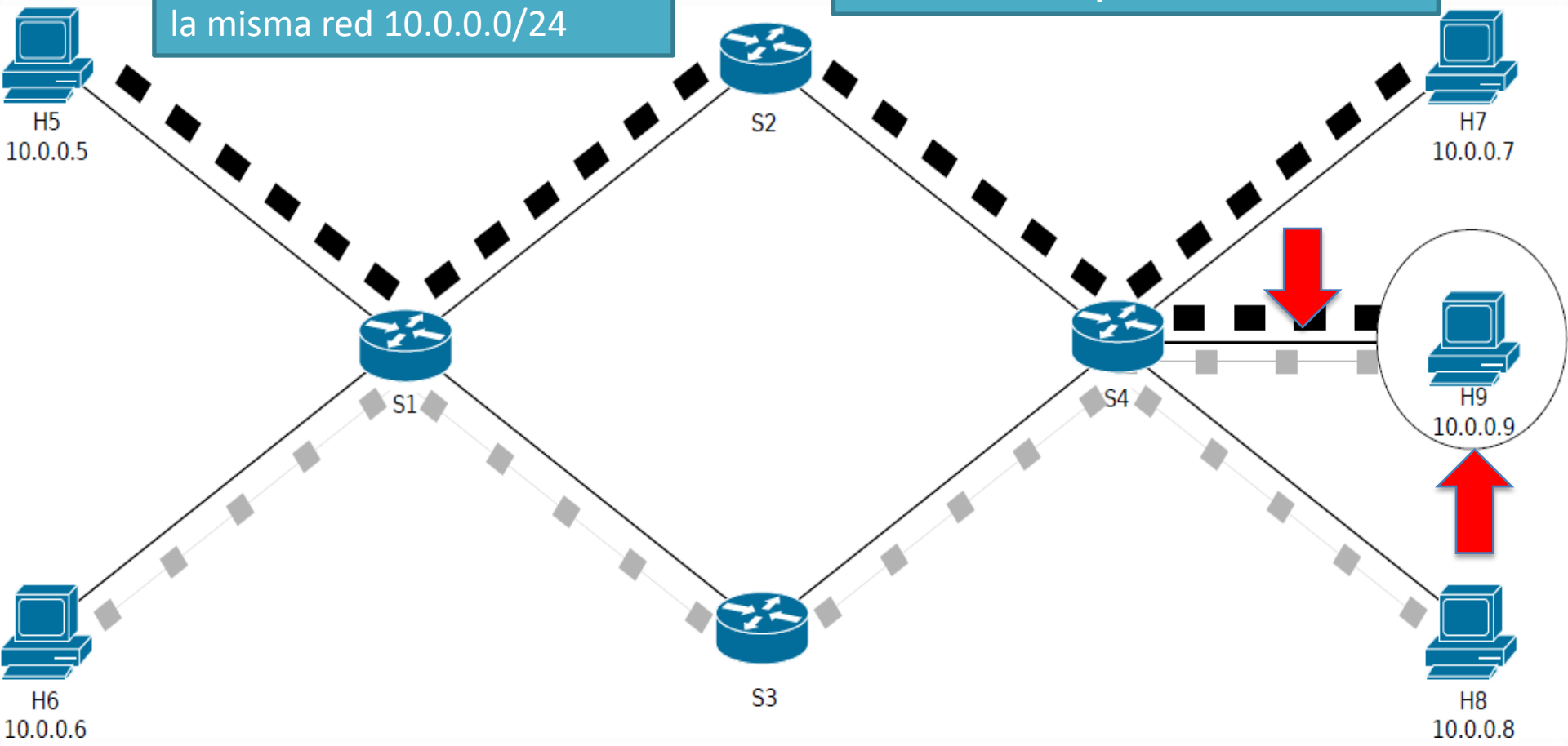
Ejemplo con Redes Sobrepuestas

- Se entiende por redes sobrepuestas por la superposición de las redes virtuales en la misma infraestructura física
  - Para lograr este resultado con el uso de OpenFlow es necesario crear un conjunto de políticas definidas por los flujos.
  - Con el uso de estas políticas se espera que las redes virtuales pueden funcionar simultáneamente y aislados unos de otros

Como crear esta red con la tecnologia atual?

NB.: Todos los enderecos IP en la misma red 10.0.0.0/24

H9 es lo unico que habla con todos



Implementacion

- Para implementar esta solución solo necesitamos programar tres archivos:
  - redes.py
    - Con la descripción de la topologia
  - redes-cli
    - Con la configuración del las direcciones IP
  - pyswitch.py
    - Programa los flujos que programa cómo la red funciona

Redes.py

```
from mininet.topo import Topo, Node

class MyTopo( Topo ):
    "Simple topology example."

    def __init__( self, enable_all = True ):
        "Create custom topo."

        # Add default members to class.
        super( MyTopo, self ).__init__()

        # Set Node IDs for hosts and switches
        leftSwitch = 1
        middleSwitchUp = 2
        middleSwitchDown = 3
        rightSwitch = 4

        leftHostUp = 5
        leftHostDown = 6
        rightHostUp = 7
        rightHostDown = 8
```

Redes.py

```
        # Add nodes
        self.add_node( leftSwitch, Node( is_switch=True ) )
        self.add_node( middleSwitchUp, Node( is_switch=True ) )
        self.add_node( middleSwitchDown, Node( is_switch=True ) )
        self.add_node( rightSwitch, Node( is_switch=True ) )
        self.add_node( leftHostUp, Node( is_switch=False ) )
        self.add_node( leftHostDown, Node( is_switch=False ) )
        self.add_node( rightHostUp, Node( is_switch=False ) )
        self.add_node( rightHostDown, Node( is_switch=False ) )

        # Add edges
        self.add_edge( leftHostUp, leftSwitch )
        self.add_edge( leftHostDown, leftSwitch )
        self.add_edge( leftSwitch, middleSwitchUp )
        self.add_edge( leftSwitch, middleSwitchDown )
        self.add_edge( middleSwitchUp, rightSwitch )
        self.add_edge( middleSwitchDown, rightSwitch )
        self.add_edge( rightSwitch, rightHostUp )
        self.add_edge( rightSwitch, rightHostDown )

        # Consider all switches and hosts 'on'
        self.enable_all()
topos = { 'mytopo': ( lambda: MyTopo() ) }
```

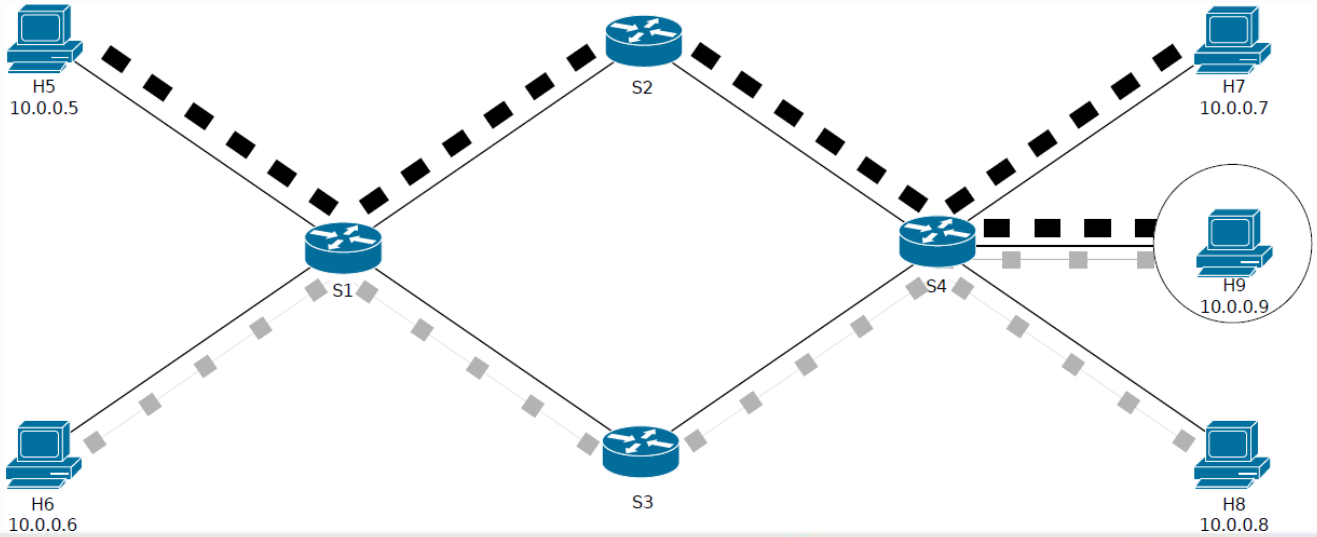
Implementação (Pyswitch.py)

```
def datapath_join_callback(dpid, stats):
    logger.info('Switch %x has joined the network' % dpid)

    if dpid == 1:
        inst.install_datapath_flow(dpid,
                                   { core.IN_PORT : 1 },
                                   openflow.OFP_FLOW_PERMANENT,
                                   openflow.OFP_FLOW_PERMANENT,
                                   [[openflow.OFPAT_OUTPUT, [0, 3]]],
                                   priority=1)
        inst.install_datapath_flow(dpid,
                                   { core.IN_PORT : 3 },
                                   openflow.OFP_FLOW_PERMANENT,
                                   openflow.OFP_FLOW_PERMANENT,
                                   [[openflow.OFPAT_OUTPUT, [0, 1]]],
                                   priority=1)
  (...)
```

Resultados

```
mininet> pingall
*** Ping: testing ping reachability
h5 -> X h7 X h9
h6 -> X X h8 h9
h7 -> h5 X X h9
h8 -> X h6 X h9
h9 -> h5 h6 h7 h8
*** Results: 40% dropped (8/20 lost)
```



Los problemas encontrados

- La falta de documentación
- Principalmente de las clases, objetos y su uso
- Python - la incapacidad de nuestro propio grupo ☺
- Limitaciones en el C + + y bibliotecas Java?
- Dificultad de depuración
- Mininet
  - Hay cosas que no aíslan bien:
    - MAC de aprendizaje en las diferentes redes nos PCs emulados
    - Fluye en los conmutadores no se detienen - Son parados en las máquinas (firewall? Comprobado con tcpdump)
  - Falta implementación de módulos listos y fácil de usar
    ARP, RARP, VLANs, QinQ**, IPv6**, encaminamiento (OSPF, BGP)*

Conclusion

- Es posible identificar la aplicabilidad directa en redes complejas usando SDN
  - Tratamiento capa dos (L2)
    - Taducción de VLANs
    - Spanning Tree (STP)
    - Agregación de enlaces (LAG)
  - Redes Moviles ☺
  - Y muchas otras…

Referências

- Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N., and Shenker, S. (2008). Nox: towards an operating system for networks.
- SIGCOMM Comput. Commun. Rev., 38:105–110.
- Lantz, B., Heller, B., and McKeown, N. (2010). A network in a laptop: rapid prototyping for software-defined networks. In Proceedings of the Ninth ACM SIGCOMMWorkshop on Hot Topics in Networks, Hotnets ’10, pages 19:1–19:6, New York, NY, USA. ACM.
- McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J. (2008). Openflow: enabling innovation in campus networks. SIGCOMM Comput. Commun. Rev., 38:69–74.

GT-MOF